Available online at www.IJournalSE.org

Emerging Science Journal

Vol. 3, No. 2, April, 2019

Sparse Nonlinear Feature Selection Algorithm via Local Structure Learning

Jiaye Li [a*], Guoqiu Wen [a], Jiangzhang Gan [a], Leyuan Zhang [a], Shanwen Zhang [a]

[a] *Guangxi Normal University, Guilin 541004, China*

Abstract

In this paper, we propose a new unsupervised feature selection algorithm by considering the nonlinear and similarity relationships within the data. To achieve this, we apply the kernel method and local structure learning to consider the nonlinear relationship between features and the local similarity between features. Specifically, we use a kernel function to map each feature of the data into the kernel space. In the high-dimensional kernel space, different features correspond to different weights, and zero weights are unimportant features (e.g. redundant features). Furthermore, we consider the similarity between features through local structure learning, and propose an effective optimization method to solve it. The experimental results show that the proposed algorithm achieves better performance than the comparison algorithm.

Keywords:

Feature Selection;
Kernel Function;
Local Structure Learning;
High-Dimensional.

Article History:

Received:	28	February	2019
Accepted:	07	April	2019

1- Introduction

With the advent of the digital age, a large amount of data has been generated. These data is difficult to deal with in terms of human capabilities and efficiency [1]. At this point, data mining and machine learning are born. In many learning tasks in machine learning (e.g., classification, clustering et al.), it is often difficult to process high-dimensional [2] data. Moreover, in many cases, the processing time and amount of calculation of the model are greatly improved when processing high-dimensional data. Therefore, it is necessary to use the feature selection algorithm [3-5] to preprocess (i.e., dimensionality reduction) high-dimensional data.

Feature selection is the selection of a subset of features that represent the overall characteristics of the data [6, 7]. In the existing feature selection algorithms, they often apply different weights to different features through a sparse regularization factor, thereby selecting relatively important features [26-29]. For example, Zhu proposes Local and Global Structure Preservation for Robust Unsupervised Spectral Feature Selection, which maintains the local structure of features by feature self-expression and maintains a global representation of features by low rank representation of sparse regularization factors. This method also uses local structure learning to maintain the local structure of the sample points. However, there are still some limitations to the existing feature selection algorithms that need to be addressed.

First, there are some feature selection algorithms that take into account the similarity between sample points, although it can take into account the local structural information of the data to some extent. But it has not been applied to the features, which is a small flaw. After considering the similarity of features, we can find that among some similar features, only a part of the features can be selected. Because it is clear that in similar features, the probability of having redundant features is large. Secondly, according to the kernel function in SVM, we can know that in low-dimensional space, only the linear relationship between features can be mined. When the features are mapped into a high-dimensional space, the

* **CONTACT**: Jiaye_ligxnu@126.com

DOI: http://dx.doi.org/10.28991/esj-2019-01175

nonlinear relationship between the features is linearly separable in the high-dimensional space. In this way, the relationship within the feature can be more fully explored, thus making the mining more thorough.

This paper proposes a new unsupervised feature selection algorithm to deal with the above two problems. Specifically, for the nonlinearity of the feature, we use a Gaussian kernel function to map each feature to a kernel matrix and then apply a weight to each feature. The important feature pairs have a large weight value, and the relatively unimportant feature (i.e., redundant feature) corresponds to a weight value of zero. For the similarity of features, we use local structure learning to consider the similarity between features. At the same time, the local structure of the feature can be maintained. For our proposed algorithm, we adopt the method of alternating iterative optimization, so that the algorithm is gradually decremented in each iteration, and finally reaches convergence. The main contributions we have listed for our approach are as follows.

- The similarity and non-linear relationship between features are considered simultaneously on a framework. The relationship between features is more fully explored through kernel sparse learning and local structure learning, thus effectively removing redundant features.
- The global structure of the data is maintained by low rank constraints, and complements the local structure learning. At the same time, low rank constraints can also remove noise samples and outliers, which improves the robustness of the algorithm to some extent.
- We propose an algorithm of alternating iterative optimization to solve our objective function. The method gradually reduces the value of the objective function in each iteration and finally converges. At the same time, we also use theory to prove the convergence of our proposed algorithm. The experimental results also show that compared with other comparison algorithms, our proposed algorithm achieves better results on real data sets.

2- Related Work

Feature selection is an important way of data pre-processing. It mainly looks for a subset of features that represent the original data. Due to the popularity of sparse learning, most of the existing feature selection algorithms use sparse learning. However, from the perspective of machine learning, the existing feature selection algorithms are mainly divided into unsupervised feature selection algorithm, semi-supervised feature selection algorithm and supervised feature selection algorithm.

In the unsupervised feature selection algorithm of the past two years, Almusallam et al. proposed an efficient unsupervised feature selection for streaming features [8]. The algorithm uses the K-means algorithm to aggregate features that are not known into a feature stream. It uses three independent similarity measures to determine whether to add existing features to the subset features by calculating the similarity measure. Wan et al. proposed global and intrinsic geometric structure embedding for unsupervised feature selection [11]. This method takes into account the information difference between the original feature space and the low-dimensional subspace. The projection matrix is constrained by the l_2 or $l_{2,1}$ − norm to select more sparse and more discriminative properties.

At the same time, some new feature selection algorithms have been proposed in the last two years. For example, Xue et al. proposed online weighted multi-task feature selection [9]. This paper proposes a weighted multitasking model that not only selects important features, but also sparse solutions. At the same time, the convergence speed of the algorithm is also guaranteed. Liu et al. proposed global and local structure preservation for feature selection [10]. The article mainly states that it is extremely important to maintain the global similar structure and local geometry of the data for supervised feature selection algorithms. For the unsupervised feature selection algorithm, it is more important to maintain the local geometry of the data. Li et al. proposed a stable feature selection algorithm [12]. The algorithm is a stable feature selection algorithm based on energy learning. It uses l_1 or l_2 regularization term to investigate its stability. Tsagris et al. proposed feature selection for high-dimensional temporal data [13]. The algorithm extends the constraint based feature selection algorithm for high-dimensional temporal data, and finally achieves good results.

There are some interesting feature selection algorithms. For example, Zhao et al. proposed cost-sensitive feature selection based on adaptive neighborhood granularity with multi-level confidence [14]. The algorithm establishes a fast backtracking based on the accuracy of the data, and designs an adaptive domain rough set model. A trade-off between test cost and misclassification cost to select useful features. Wang et al. proposed category specific dictionary learning for attribute specific feature selection [15]. This article proposes a method of combining label learning with dictionary learning. Feature selection is implemented at the dictionary level, which can better preserve structural information. At the same time, the intra-class noise is suppressed. Sheeja et al. proposed a novel feature selection method using fuzzy rough sets [16]. The algorithm studies the properties of fuzzy rough approximations, using the divergence metrics of fuzzy sets to define, and using fuzzy measures to express the different fuzzy sets. Zhang et al. proposed a fast feature selection algorithm based on swarm intelligence in acoustic defect detection [17]. The algorithm transforms the feature selection into a global optimization problem, and proposes a filter-based feature global optimization framework and mathematical model. Finally, the algorithm takes the shortest time in the case of equal performance.

3- Our Method

In this section, we first introduce the symbols used in this article and then explain our proposed Unsupervised Feature Selection Algorithm via Local Structure Learning and Kernel Function(Abbreviated as: LSK FS) , in Sections 3.1 and 3.2, respectively, and then optimize the proposed optimization method in Section 3.3. Finally, we analysis the convergence of the objective function in Section 3.4.

3-1- Notations

For the data matrix $\boldsymbol{X} \in \boldsymbol{R}^{n\times d}$, the *i*-th row and the *j*-th column are denoted as $\boldsymbol{X}^i$ and $\boldsymbol{X}_j$ respectively, and the elements of the *i*-th row and the *j*-th column are denoted as $x_{i,j}$. The trace of the matrix $\boldsymbol{X}$ is denoted by $tr(\boldsymbol{X})$, $\boldsymbol{X}^T$ denotes the transpose of the matrix $\boldsymbol{X}$, and $\boldsymbol{X}^{-1}$ represents the inverse of the matrix $\boldsymbol{X}$. We also denote the norm and norm of $\boldsymbol{X}$ respectively as $\|\boldsymbol{X}\|_F = \sqrt{\sum_i^n \|x^i\|_2^2} = \sqrt{\sum_j^d \|x_j\|_2^2}$ and $\|X\|_1 = \sum_{j=1}^d |x_j|$.

3-2- LSK FS Algorithm

Suppose a given sample data set $\boldsymbol{X} \in \boldsymbol{R}^{n\times d}$, where *n* and *d* represent the number of samples and the number of attributes, respectively.

This paper first breaks the data set $\boldsymbol{X} \in \boldsymbol{R}^{n\times d}$ into *d* column vectors, each vector $x_i \in R^{n\times 1}, i = 1,...,d$. Then it treats each element in each x_i as an independent feature value $x_{ij} \in R, j = 1,...,n$. And projects them into the kernel space to get the kernel matrix $\boldsymbol{K}^{(i)} \in \boldsymbol{R}^{n\times n}$, namely:

$$\boldsymbol{K}^{(i)} = \begin{bmatrix} k(x_{i1},x_{i1}) & k(x_{i1},x_{i2}) & \dots & k(x_{i1},x_{in}) \\ k(x_{i2},x_{i1}) & k(x_{i2},x_{i2}) & \dots & k(x_{i2},x_{in}) \\ \dots & \dots & \dots & \dots \\ k(x_{in},x_{i1}) & k(x_{in},x_{i2}) & \dots & k(x_{in},x_{in}) \end{bmatrix} \tag{1}$$

Thus the original $\boldsymbol{X} \in \boldsymbol{R}^{n\times d}$ becomes *d* kernel matrices.

The unsupervised feature selection algorithm mainly mines more representative features in the data. In the absence of the class label $\boldsymbol{Y}$, using the data matrix $\boldsymbol{X}$ as a response matrix, the internal structure of the original features of the data can be better preserved [18, 19]. In order to fully exploit the nonlinear relationship of data features. Get the following expression:

$$\boldsymbol{X} = \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{W} \tag{2}$$

Where $\boldsymbol{W} \in \boldsymbol{R}^{n\times d}$ represents the kernel coefficient matrix, $\boldsymbol{\alpha} \in \boldsymbol{R}^{d\times 1}$ is used to perform feature selection, which is equivalent to the weight vector of the feature, α_i is an element of the vector $\boldsymbol{\alpha}$, $\boldsymbol{K}^{(i)} \in \boldsymbol{R}^{n\times n}$ is the kernel matrix.

Predecessors have proved that the local structure between data can be used to reduce the dimension [20], so this paper makes local structure learning by establishing a similarity matrix between data features in low-dimensional space. The following formula is obtained through local structure learning:

$$\min_{\boldsymbol{z}} \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{W} - x_j^T \boldsymbol{W} \right\|_2^2 s_{i,j} \tag{3}$$

Where $x_i \in R^{n\times 1}$ represents *i*-th feature. $\boldsymbol{W} \in \boldsymbol{R}^{n\times d}$ is the conversion matrix of high-dimensional data in low-dimensional space, $s_{i,j}$ is an element of matrix $\boldsymbol{S}$, indicating the similarity between feature x_i and feature x_j . If the feature x_i is the *k*-th nearest neighbor of the feature x_j , then the value $s_{i,j}$ is obtained by the Gaussian kernel function; otherwise =0. In order to make $\boldsymbol{X}$ get a better fitting effect, and consider the structural relationship between data features in low-dimensional space, we get the following formula:

$$\min_{\boldsymbol{S},\boldsymbol{W},\boldsymbol{\alpha}} \left\| \boldsymbol{X} - \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{W} \right\|_F^2 + \lambda_1 \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{W} - x_j^T \boldsymbol{W} \right\|_2^2 s_{i,j} \tag{4}$$

Since the similarity matrix $\boldsymbol{S}$ is particularly affected by the influence of parameters σ . In order to reduce the number of adjustment parameters, a more efficient similarity matrix is learned. In this paper, structural learning and low-dimensional space learning are alternated to achieve their optimal results. Specifically get the following formula:

$$\begin{aligned} &\min_{\boldsymbol{S},\boldsymbol{W},\boldsymbol{\alpha}} \left\| \boldsymbol{X} - \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{W} \right\|_F^2 + \lambda_1 \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{W} - x_j^T \boldsymbol{W} \right\|_2^2 s_{i,j} + \lambda_2 \left\| s_i \right\|_2^2 \\ &s.t., \forall i, s_i^T \mathbf{1} = 1, s_{i,i} = 0, s_{i,j} \geq 0, \text{if } j \in \mathrm{N}(i), otherwise\ 0 \end{aligned} \tag{5}$$

Where λ_1 , λ_2 is the tuning parameter, s_i is the *i*-th column of the similar matrix $\boldsymbol{S}$, and $\|s_i\|_2^2$ is used to avoid unimportant results. **1** represents a vector with all elements of 1. $\mathrm{N}(i)$ represents a set of neighbors the *i*-th feature. In

order to maintain rotation invariance, we set $s_i^T \mathbf{1} = 1$. Therefore, the above formula can make the similarity corresponding to the feature having a relatively close distance large, and the feature having a relatively long distance has a small corresponding value.

In order to eliminate the interference of the outliers, the noise samples are removed at the same time [21]. This paper adds a low rank constraint [22] to the matrix $\boldsymbol{W}$, namely:

$$\boldsymbol{W} = \boldsymbol{AB} \tag{6}$$

Where $\boldsymbol{A} \in \boldsymbol{R}^{m \times r}$, $\boldsymbol{B} \in \boldsymbol{R}^{r \times d}$, $r \le \min(n, d)$, we added an orthogonal limit to $\boldsymbol{A}$, in order to fully consider the correlation between the output variables. We add a l_1 – norm of $\boldsymbol{\alpha}$ for sparse learning and feature selection [23]. Finally we get our final objective function as follows:

$$\begin{aligned} &\min_{\boldsymbol{S},\boldsymbol{A},\boldsymbol{B},\boldsymbol{\alpha}} \left\| \boldsymbol{X} - \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{AB} \right\|_F^2 + \lambda_1 \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{AB} - x_j^T \boldsymbol{AB} \right\|_2^2 s_{i,j} \\ &+ \lambda_2 \left\| s_i \right\|_2^2 + \lambda_3 \left\| \boldsymbol{\alpha} \right\|_1 \\ &s.t., \forall i, s_i^T \mathbf{1} = 1, s_{i,i} = 0, \\ &s_{i,j} \ge 0, \text{if } j \in \mathrm{N}(i), otherwise\ 0, \boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{I} \end{aligned} \tag{7}$$

Where $\boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{I} \in \boldsymbol{R}^{r \times r}$, λ_1, λ_2 and λ_3 are the tuning parameter. The kernel matrix $\boldsymbol{K}$ is calculated by the Gaussian kernel function, and its main function is to map the data to the kernel space, thereby mining the nonlinear relationship between the data features. The l_1 – norm of the last item $\boldsymbol{\alpha}$ is used to sparse the features for feature selection. If the value of the element corresponding to the vector $\boldsymbol{\alpha}$ is zero, it means that the feature is not selected.

3-3- Optimization

Since the objective function is not co-convex, the closed solution cannot be directly obtained. Therefore, this paper proposes an alternate iterative optimization method to solve the problem, which is divided into the following four steps:

Update $\boldsymbol{A}$ by fixing $\boldsymbol{S}$, $\boldsymbol{\alpha}$ and $\boldsymbol{B}$:

When $\boldsymbol{S}$, $\boldsymbol{\alpha}$ and $\boldsymbol{B}$ are fixed, the optimization (7) problem becomes:

$$\begin{aligned} \min_{\boldsymbol{A}} &\left\| \boldsymbol{X} - \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{AB} \right\|_F^2 + \lambda_1 \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{AB} - x_j^T \boldsymbol{AB} \right\|_2^2 s_{i,j} \\ &s.t., \boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{I} \end{aligned} \tag{8}$$

We make $\boldsymbol{P} = \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)}$, then the (8) formula can be transformed into:

$$\begin{aligned} \min_{\boldsymbol{A}} &\left\| \boldsymbol{X} - \boldsymbol{PAB} \right\|_F^2 + \lambda_1 \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{AB} - x_j^T \boldsymbol{AB} \right\|_2^2 s_{i,j} \\ &s.t., \boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{I} \end{aligned} \tag{9}$$

We simplify the (9), we have:

$$\begin{aligned} &\min_{\boldsymbol{A}} tr(\boldsymbol{X}^T \boldsymbol{X} - \boldsymbol{X}^T \boldsymbol{PAB} - \boldsymbol{B}^T \boldsymbol{A}^T \boldsymbol{P}^T \boldsymbol{X} + \boldsymbol{B}^T \boldsymbol{A}^T \boldsymbol{P}^T \boldsymbol{PAB}) \\ &+ \lambda_1 tr(\boldsymbol{B}^T \boldsymbol{A}^T \boldsymbol{XLX}^T \boldsymbol{AB}), s.t., \boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{I} \end{aligned} \tag{10}$$

Where $\mathrm{tr}(\cdot)$ represents the trace of matrix, $\boldsymbol{L} = \boldsymbol{Q} - \boldsymbol{S} \in \boldsymbol{R}^{d \times d}$ is a Laplace matrix, $\boldsymbol{Q}$ is a diagonal matrix, and the elements of each column are $q_{i,i} = \sum_{j=1}^{d} s_{i,j}$. Deriving for $\boldsymbol{A}$, we have:

$$2\lambda_1 \boldsymbol{XLX}^T \boldsymbol{ABB}^T - 2\boldsymbol{P}^T \boldsymbol{XB}^T + 2\boldsymbol{P}^T \boldsymbol{PABB}^T \tag{11}$$

Due to the orthogonality of $\boldsymbol{A}$, we can optimize it by the method in [24].

Update $\boldsymbol{B}$ by fixing $\boldsymbol{S}$, $\boldsymbol{\alpha}$ and $\boldsymbol{A}$

By fixing $\boldsymbol{S}$, $\boldsymbol{\alpha}$ and $\boldsymbol{A}$, the objective function (7) can be simplified as follows:

$$\min_{\boldsymbol{B}} \left\| \boldsymbol{X} - \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{AB} \right\|_F^2 + \lambda_1 \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{AB} - x_j^T \boldsymbol{AB} \right\|_2^2 s_{i,j} \tag{12}$$

It is easy to get (12) is equivalent to the following formula:

$$\begin{aligned} &\min_{\boldsymbol{B}} tr(\boldsymbol{X}^T \boldsymbol{X} - \boldsymbol{X}^T \boldsymbol{PAB} - \boldsymbol{B}^T \boldsymbol{A}^T \boldsymbol{P}^T \boldsymbol{X} + \boldsymbol{B}^T \boldsymbol{A}^T \boldsymbol{P}^T \boldsymbol{PAB}) \\ &+ \lambda_1 tr(\boldsymbol{B}^T \boldsymbol{A}^T \boldsymbol{XLX}^T \boldsymbol{AB}) \end{aligned} \tag{13}$$

When we ask for $\boldsymbol{B}$ and let its derivative be zero, we can get:

$$\boldsymbol{B} = (\boldsymbol{A}^T \boldsymbol{P}^T \boldsymbol{PA} + \lambda_1 \boldsymbol{A}^T \boldsymbol{XLX}^T \boldsymbol{A})^{-1} \boldsymbol{A}^T \boldsymbol{P}^T \boldsymbol{X} \tag{14}$$

Update S by fixing *A*, $\boldsymbol{\alpha}$ and *B*

After fixing ***A***, $\boldsymbol{\alpha}$ and ***B***, the objective function (7) becomes:

$$\begin{aligned}&\min_{\boldsymbol{S}} \lambda_1 \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{AB} - x_j^T \boldsymbol{AB} \right\|_2^2 s_{i,j} + \lambda_2 \left\| s_i \right\|_2^2 \\ &s.t., \forall i, s_i^T \mathbf{1} = 1, \mathrm{s}_{i,i} = 0, \\ &s_{i,j} \geq 0, \text{if } j \in \mathrm{N}(i), otherwise\ 0\end{aligned} \tag{15}$$

We first calculate the Euclidean distance between every two data features to construct the neighbours of all the features. If the *j*-th feature does not belong to the nearest neighbour of the *i*-th feature, then the value of $s_{i,j}$ is zero; otherwise, the value of $s_{i,j}$ is solved by equation (18).

At the same time, optimizing ***S*** is equivalent to optimizing each $s_i (i = 1, ..., d)$ individually, so we further translate the optimization problem into the following equation:

$$\min_{s_i^T 1 = 1, s_{i,i} = 0, s_{i,j} \geq 0} \sum_{i,j}^{d} (\lambda_1 \left\| x_i^T \boldsymbol{AB} - x_j^T \boldsymbol{AB} \right\|_2^2 s_{i,j} + \lambda_2 s_{i,j}^2) \tag{16}$$

Here, $\boldsymbol{Z} \in \boldsymbol{R}^{d \times d}$, in which $\boldsymbol{Z}_{i,j} = \lambda_1 \left\| x_i^T \boldsymbol{AB} - x_j^T \boldsymbol{AB} \right\|_2^2$, such (16) further becomes:

$$\min_{s_i^T 1 = 1, s_{i,i} = 0, s_{i,j} \geq 0} \left\| s_i + \frac{\lambda_1}{2\lambda_2} \boldsymbol{Z}_i \right\|_2^2 \tag{17}$$

Under KKT conditions, we can get the following:

$$s_{i,j} = (-\frac{\lambda_1}{2\lambda_2} Z_{i,j} + \tau)_+ \tag{18}$$

Since each data feature has a neighbour, we sort each $Z_i (i = 1, ...d)$ in descending order, that is $\hat{Z}_i = \{\hat{Z}_{i,1}, ..., \hat{Z}_{i,d}\}$, we know: $s_{i,k+1} = 0$, $s_{i,k} > 0$. We have:

$$-\frac{\lambda_1}{2\lambda_2} \hat{Z}_{i,k+1} + \tau \leq 0 \tag{19}$$

Under the conditions $s_i^T \mathbf{1} = 1$, we can get:

$$\sum_{j=1}^{k} (\frac{\lambda_1}{2\lambda_2} \hat{Z}_{i,k} + \tau) = 1 \Rightarrow \tau = \frac{1}{k} + \frac{\lambda_1}{2k\lambda_2} \sum_{j=1}^{k} \hat{Z}_{i,k} \tag{20}$$

Update $\boldsymbol{\alpha}$ by fixing *A*, B and *S*

After fixing ***A***, ***B*** and ***S***, the objective function (7) becomes:

$$\min_{\boldsymbol{\alpha}} \left\| \boldsymbol{X} - \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{AB} \right\|_F^2 + \lambda_3 \left\| \boldsymbol{\alpha} \right\|_1 \tag{21}$$

In order to optimize the next step, here is the simplification of the above formula, namely:

$$\begin{aligned}&\Leftrightarrow \min_{\boldsymbol{\alpha}} \left\| \boldsymbol{X} - (\alpha_1 \boldsymbol{K}^{(1)} \boldsymbol{AB} + ... + \alpha_d \boldsymbol{K}^{(d)} \boldsymbol{AB}) \right\|_F^2 + \lambda_3 \left\| \boldsymbol{\alpha} \right\|_1 \\ &\overset{\boldsymbol{Q}^{(i)} = \boldsymbol{K}^{(i)} \boldsymbol{AB} \in \boldsymbol{R}^{n \times d}}{\Leftrightarrow} \min_{\boldsymbol{\alpha}} \left\| \boldsymbol{X} - (\alpha_1 \boldsymbol{Q}^{(1)} + ... + \alpha_d \boldsymbol{Q}^{(d)}) \right\|_F^2 + \lambda_3 \left\| \boldsymbol{\alpha} \right\|_1 \\ &\Leftrightarrow \min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \left\| \boldsymbol{X}_i - (\alpha_1 q_{i,\cdot}^{(1)} + ... + \alpha_d q_{i,\cdot}^{(d)}) \right\|_2^2 + \lambda_3 \left\| \boldsymbol{\alpha} \right\|_1\end{aligned} \tag{22}$$

We set $\boldsymbol{M}^{(i)} = \begin{pmatrix} q_{i,1}^{(1)} & q_{i,d}^{(1)} \\ & \\ q_{i,1}^{(d)} & q_{i,d}^{(d)} \end{pmatrix} \in \boldsymbol{R}^{d \times d}$ and have:

$$\begin{aligned}&\Leftrightarrow \min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \left\| \boldsymbol{X}_i - \boldsymbol{\alpha}^T \boldsymbol{M}^{(i)} \right\|_2^2 + \lambda_3 \left\| \boldsymbol{\alpha} \right\|_1 \\ &\Leftrightarrow \min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \left\| \boldsymbol{X}_i^T - (\boldsymbol{M}^{(i)})^T \boldsymbol{\alpha} \right\|_2^2 + \lambda_3 \left\| \boldsymbol{\alpha} \right\|_1 \\ &\Leftrightarrow \min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \boldsymbol{X}_i \boldsymbol{X}_i^T - 2\boldsymbol{\alpha}^T \sum_{i=1}^{n} \boldsymbol{M}^{(i)} \boldsymbol{X}_i^T \\ &+ \boldsymbol{\alpha}^T \sum_{i=1}^{n} (\boldsymbol{M}^{(i)} (\boldsymbol{M}^{(i)})^T) \boldsymbol{\alpha} + \lambda_3 \left\| \boldsymbol{\alpha} \right\|_1\end{aligned} \tag{23}$$

The above simplification is only for the convenience of the following gradient descent [25]. We make:

$$f(\boldsymbol{\alpha})=\left\|\boldsymbol{X}-\sum_{i=1}^{d}\alpha_i\boldsymbol{K}^{(i)}\boldsymbol{AB}\right\|_F^2$$
$$F(\boldsymbol{\alpha})=f(\boldsymbol{\alpha})+\lambda_3\|\boldsymbol{\alpha}\|_1 \tag{24}$$

Note that $\|\boldsymbol{\alpha}\|_1$ is convex but not smooth. So using approximate gradient to optimize $\boldsymbol{\alpha}$, we can update iterations $\boldsymbol{\alpha}$ by the following rules.

$$\boldsymbol{\alpha}_{t+1}=\arg\min_{\boldsymbol{\alpha}} G_{\eta t}(\boldsymbol{\alpha},\boldsymbol{\alpha}_t) \tag{25}$$

$$G_{\eta t}(\boldsymbol{\alpha},\boldsymbol{\alpha}_t)=f(\boldsymbol{\alpha}_t)+<\nabla f(\boldsymbol{\alpha}_t),\boldsymbol{\alpha}-\boldsymbol{\alpha}_t>+\frac{\eta_t}{2}\|\boldsymbol{\alpha}-\boldsymbol{\alpha}_t\|^2+\lambda_3\|\boldsymbol{\alpha}\|_1 \tag{26}$$

In the above formula, $\nabla f(\boldsymbol{\alpha}_t)=2\boldsymbol{\alpha}_t^T\sum_{i=1}^{n}(\boldsymbol{M}^{(i)}(\boldsymbol{M}^{(i)})^T)-2\sum_{i=1}^{n}\boldsymbol{X}_i(\boldsymbol{M}^{(i)})^T$, η_t is a tuning parameter, $\boldsymbol{\alpha}_t$ is the value of $\boldsymbol{\alpha}$ in the *t*-th iteration.

By ignoring the independence in equation (26), we can get:

$$\boldsymbol{\alpha}_{t+1}=\pi_{\eta_t}(\boldsymbol{\alpha}_t)=\arg\min_{\alpha}\frac{1}{2}\|\boldsymbol{\alpha}-\boldsymbol{U}_t\|_F^2+\frac{\lambda_3}{\eta_t}\|\boldsymbol{\alpha}\|_1 \tag{27}$$

Where $\boldsymbol{U}_t=\boldsymbol{\alpha}_t-\frac{1}{\eta_t}\nabla f(\boldsymbol{\alpha}_t)$, $\pi_{\eta_t}(\boldsymbol{\alpha}_t)$ is the Euclidean projection on the convex set η_t, because $\|\boldsymbol{\alpha}\|_1$ has a separable form, the formula (27) can be written as follows:

$$\alpha_{t+1}^i=\arg\min_{\alpha^i}\frac{1}{2}\|\alpha^i-U_t^i\|_2^2+\frac{\lambda_3}{\eta_t}|\alpha^i| \tag{28}$$

Where α^i and α_{t+1}^i are the *i*-th elements of $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}_{t+1}$ respectively, then according to formula (28), α_{t+1}^i can obtain the following closed solution:

$$\alpha^{i*}=\begin{cases}u_t^i-\frac{\lambda_3}{\eta_t}\times sign(u_t^i), & \text{if } \|u_t^i\|>\frac{\lambda_3}{\eta_t}\\ 0, & \text{otherwise.}\end{cases} \tag{29}$$

To speed up the approximate gradient algorithm in equation (26), we have added auxiliary variables:

$$\boldsymbol{V}_{t+1}=\boldsymbol{\alpha}_t+\frac{\beta_t-1}{\beta_{t+1}}(\boldsymbol{\alpha}_{t+1}-\boldsymbol{\alpha}_t) \tag{30}$$

Where $\beta_{t+1}=\frac{1+\sqrt{1+4\beta_t^2}}{2}$.

Algorithm 1. Pseudo code for solving (7)

Input: $\boldsymbol{X}\in\boldsymbol{R}^{n\times d}$, λ_1, λ_2, λ_3, r, η_0, $\beta_1=1$, γ;
Output: $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{S}$, $\boldsymbol{\alpha}$;
1. Initialize *t*=1;
2. Randomly initialize $\boldsymbol{A}^{(0)}$, $\boldsymbol{B}^{(0)}$, and $\boldsymbol{\alpha}_0$;
3. Compute $\boldsymbol{K}^{(i)}$ by formula (1);
4. Repeat
5. Update $\boldsymbol{A}$ via formula (8);
6. Update $\boldsymbol{B}$ via formula (12);
7. Update $\boldsymbol{S}$ via formula (15);
8. Update $\boldsymbol{\alpha}$ according to the following rules
9. while $F(\boldsymbol{\alpha}_t)>G_{\eta t-1}(\pi_{\eta t-1}(\boldsymbol{\alpha}_t),\boldsymbol{\alpha}_t)$;
10. Set $\eta_{t-1}=\gamma\eta_{t-1}$;
11. end while
12. Set $\eta_t=\gamma\eta_{t-1}$;
13. Compute $\boldsymbol{\alpha}_{t+1}=\arg\min_{\boldsymbol{\alpha}} G_{\eta t}(\boldsymbol{\alpha},\boldsymbol{V}_t)$;
14. Compute $\beta_{t+1}=\frac{1+\sqrt{1+4\beta_t^2}}{2}$;
15. Compute formula(30);
16. *t*=*t*+1;
17. **until** *converge*;

3-4- Convergence Analysis

According to (18), for all $i, j = 1,...,n$, $s_{i,j}^{(t+1)}$ has a closed solution. We can get the following formula:

$$\begin{aligned}&\left\|\boldsymbol{X}-\sum_{i=1}^{d}\alpha_i\boldsymbol{K}^{(i)}\boldsymbol{A}^{(t)}\boldsymbol{B}^{(t)}\right\|_F^2+\lambda_1\sum_{i,j}^{d}\left\|x_i^T\boldsymbol{A}^{(t)}\boldsymbol{B}^{(t)}-x_j^T\boldsymbol{A}^{(t)}\boldsymbol{B}^{(t)}\right\|_2^2 s_{i,j}^{(t+1)}\\&+\lambda_2\sum_{i,j}^{d}\left\|s_i^{(t+1)}\right\|_2^2\\&\le\left\|\boldsymbol{X}-\sum_{i=1}^{d}\alpha_i\boldsymbol{K}^{(i)}\boldsymbol{A}^{(t)}\boldsymbol{B}^{(t)}\right\|_F^2+\lambda_1\sum_{i,j}^{d}\left\|x_i^T\boldsymbol{A}^{(t)}\boldsymbol{B}^{(t)}-x_j^T\boldsymbol{A}^{(t)}\boldsymbol{B}^{(t)}\right\|_2^2 s_{i,j}^{(t)}\\&+\lambda_2\sum_{i,j}^{d}\left\|s_i^{(t)}\right\|_2^2\end{aligned}\tag{31}$$

When $\boldsymbol{\alpha}$ and $\boldsymbol{S}^{(t+1)}$ are fixed, to update $\boldsymbol{A}^{(t+1)}$ and $\boldsymbol{B}^{(t+1)}$, we can get:

$$\begin{aligned}&\left\|\boldsymbol{X}-\sum_{i=1}^{d}\alpha_i\boldsymbol{K}^{(i)}\boldsymbol{A}^{(t+1)}\boldsymbol{B}^{(t+1)}\right\|_F^2+\lambda_1\sum_{i,j}^{d}\left\|x_i^T\boldsymbol{A}^{(t+1)}\boldsymbol{B}^{(t+1)}-x_j^T\boldsymbol{A}^{(t+1)}\boldsymbol{B}^{(t+1)}\right\|_2^2 s_{i,j}^{(t+1)}\\&+\lambda_2\sum_{i,j}^{d}\left\|s_i^{(t+1)}\right\|_2^2\\&\le\left\|\boldsymbol{X}-\sum_{i=1}^{d}\alpha_i\boldsymbol{K}^{(i)}\boldsymbol{A}^{(t)}\boldsymbol{B}^{(t)}\right\|_F^2+\lambda_1\sum_{i,j}^{d}\left\|x_i^T\boldsymbol{A}^{(t)}\boldsymbol{B}^{(t)}-x_j^T\boldsymbol{A}^{(t)}\boldsymbol{B}^{(t)}\right\|_2^2 s_{i,j}^{(t+1)}\\&+\lambda_2\sum_{i,j}^{d}\left\|s_i^{(t+1)}\right\|_2^2\end{aligned}\tag{32}$$

From the above two types, we have:

$$\begin{aligned}&\left\|\boldsymbol{X}-\sum_{i=1}^{d}\alpha_i\boldsymbol{K}^{(i)}\boldsymbol{A}^{(t+1)}\boldsymbol{B}^{(t+1)}\right\|_F^2+\lambda_1\sum_{i,j}^{d}\left\|x_i^T\boldsymbol{A}^{(t+1)}\boldsymbol{B}^{(t+1)}-x_j^T\boldsymbol{A}^{(t+1)}\boldsymbol{B}^{(t+1)}\right\|_2^2 s_{i,j}^{(t+1)}\\&+\lambda_2\sum_{i,j}^{d}\left\|s_i^{(t+1)}\right\|_2^2\\&\le\left\|\boldsymbol{X}-\sum_{i=1}^{d}\alpha_i\boldsymbol{K}^{(i)}\boldsymbol{A}^{(t)}\boldsymbol{B}^{(t)}\right\|_F^2+\lambda_1\sum_{i,j}^{d}\left\|x_i^T\boldsymbol{A}^{(t)}\boldsymbol{B}^{(t)}-x_j^T\boldsymbol{A}^{(t)}\boldsymbol{B}^{(t)}\right\|_2^2 s_{i,j}^{(t)}\\&+\lambda_2\sum_{i,j}^{d}\left\|s_i^{(t)}\right\|_2^2\end{aligned}\tag{33}$$

Theorem 1 let α_t be the sequence generated by Algorithm 1, then for $\forall t \ge 1$, (34) holds:

$$F(\boldsymbol{\alpha}_t)-F(\boldsymbol{\alpha}^*)\le\frac{2\gamma L\left\|\boldsymbol{\alpha}_1-\boldsymbol{\alpha}^*\right\|_F^2}{(t+1)^2}\tag{34}$$

According to reference [38], γ is a constant defined in advance, L is the Lipschitz constant of the $f(\boldsymbol{\alpha})$ gradient in equation (24), and $\boldsymbol{\alpha}^*=\arg\min_{\boldsymbol{\alpha}} F(\boldsymbol{\alpha})$.

Through the above inequality and Theorem 1, we can easily see that our algorithm is convergent.

4- Experiments

We compared the classification accuracy of our algorithm and 8 comparison algorithms in 12 data sets (shown in Table 1).

4-1- Experimental Settings

We tested our proposed unsupervised feature selection algorithm on four binary data sets and eight multi-class data sets. They are Yale, Colon, Lung_discrete, Glass, SPECTF, Sonar, Clean, Arrhythmia, Movements, Ecoli, Urban_land and Forest, where the first three data sets are from feature selection data, and the last nine data sets are from the UCI data set. The details of the data set are shown in Table 1:

Table1. The information of the data sets

Datasets	Samples	Dimensions	Classes
Glass	214	9	6
Movements	360	90	15
SPECTF	267	44	2
Ecoli	336	343	8
Sonar	208	60	2
Urban_land	168	147	9

Clean	476	167	2
Forest	325	27	4
Arrhythmia	452	279	13
Colon	62	2000	2
Yale	165	1024	15
Lungdiscrete	73	325	7

At the same time, we found eight representative feature selection comparison algorithms to compare with our proposed algorithm. The main introduction of the algorithm is as follows:

EUFS [30]: This paper proposes a new unsupervised feature selection algorithm, which is different from other unsupervised feature selection algorithms to generate tags through clustering algorithms. This method directly embeds feature selection into the clustering algorithm through sparse learning theory. The most prominent contribution of this method is that other unsupervised feature selection algorithms can be applied to this framework.

FSASL [31]: The intrinsic structure of the data is rarely considered for existing feature selection algorithms. By placing structural learning and feature selection in a framework at the same time, the algorithm can effectively select representative features while maintaining the structure of the data (i.e., structural learning and feature selection complement each other).

NDFS [32]: The algorithm is a new unsupervised feature selection algorithm that mines discerning information. Specifically, it imposes a non-negative constraint on the class indication to learn clustering tags more accurately. At the same time, $l_{2,1}$ – norm is used to remove redundant features. It is an algorithm that performs clustering and feature selection simultaneously.

NetFS [33]: This method is also an unsupervised feature selection algorithm, but the algorithm is mainly for Networked Data. Due to the large amount of noise in Networked Data, the algorithm combines latent representation learning for feature selection. Through sparse learning and latent representation learning, the algorithm can remove the noise interference well and finally achieve good robustness.

RLSR [34]: This paper proposes a semi-supervised feature selection algorithm by finding the global and sparse solutions of the projection matrix. The main contribution of the algorithm is to propose a regular term $l_{2,1}^{2}$ – norm, and gives a detailed theoretical proof, which proves that the regular term can effectively select features and take into account global information.

RFS [35]: This paper presents a new robust feature selection algorithm through sparse learning. Specifically, it applies $l_{2,1}$ – norm to both the loss function and the regular item. Sparse restriction on the regular term can effectively remove the redundant features. The $l_{2,1}$ – norm of the loss function can effectively remove the noise samples, thus achieving a robust effect.

RSR [36]: The algorithm is a new unsupervised feature selection algorithm that learns by regularized self-characterization. Specifically, if a feature is particularly important, it can be represented linearly by most other features (i.e., a feature can be represented by a combination of other features). In addition, the algorithm shows good performance on both manual and real data sets.

K_OFSD [37]: The algorithm is an online feature selection algorithm, which mainly selects relevant features through neighbor learning. Through this method, the class imbalance problem can be effectively solved. At the same time, the dimensionality of the high-dimensional data is effectively reduced by the dependency between the conditional features and the decision-making class.

In our proposed model, we set $\{\lambda_1, \lambda_2\} \in \{10^{-2}, ..., 10^{2}\}$, the rank of the kernel coefficient matrix $r \in \{1, ..., \min(n, d)\}$, and the parameter of $l_{2,p}$ – norm $p \in \{0.1, ..., 1.9\}$. The parameters $c \in \{2^{-5}, ..., 2^{5}\}$ and $g \in \{2^{-5}, ..., 2^{5}\}$ are used to select the best SVM for classification. Through 10-fold cross-validation, we divide the data set into a training set and a test set. In order to minimize the experimental error, we perform 10 times experiments, and finally find the average of the classification accuracy.

Since the experiment uses the classification accuracy rate to measure the performance of the algorithm, we define the classification accuracy as follows:

$$acc = X_{correct} / X \tag{31}$$

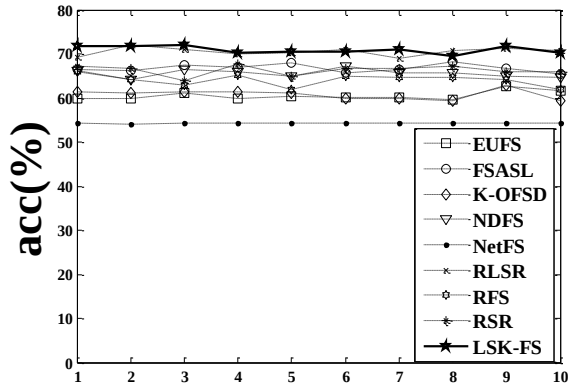
Where X represents the total number of samples and $X_{correct}$ represents the correct number of samples for classification. At the same time, we define the standard deviation to measure the stability of our algorithm, as follows:

$$std = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(acc_i - \mu)^2} \tag{32}$$

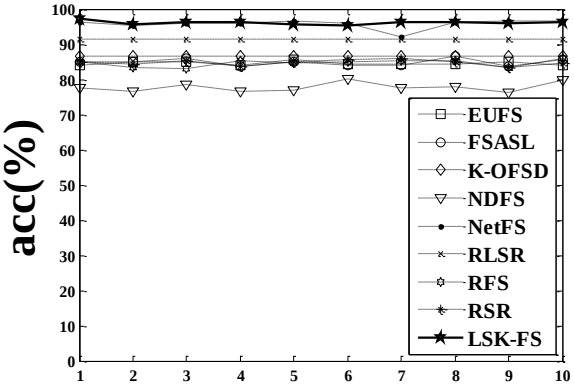
Where N represents the number of experiments, acc_i represents the classification accuracy of the i-th experiment, μ represents the average classification accuracy, and the smaller the std, the more stable the representative algorithm.

4-2- Experiment Result

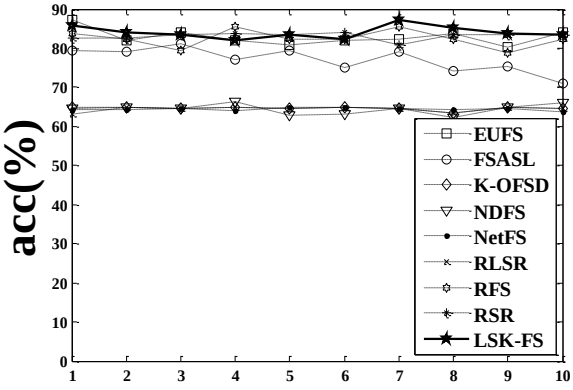
In Figure 1, we can clearly see the classification accuracy of the 10 experiments. The algorithm we proposed is not the highest every time, but most of the cases are the highest. In Table 2, we can see the average classification accuracy of each algorithm on 12 data sets. The algorithm proposed by us is obviously superior to other comparison algorithms. Specifically, it is 4.78% higher than EUFS in average classification accuracy and 5.05% higher than FSASL, which indicates that our algorithm is better than the general feature selection algorithm. Compared with K_OFSD, NDFS, NetFS, RLSR, RFS, and RSR. LSK_FS increased 13.63%, 8.55%, 6.69%, 7.88%, 6.68%, and 3.59%, respectively. In particular, our algorithm is particularly effective on the dataset SPECTF.



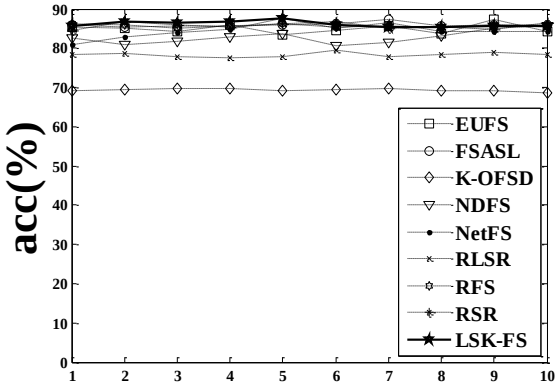
Arrhythmia



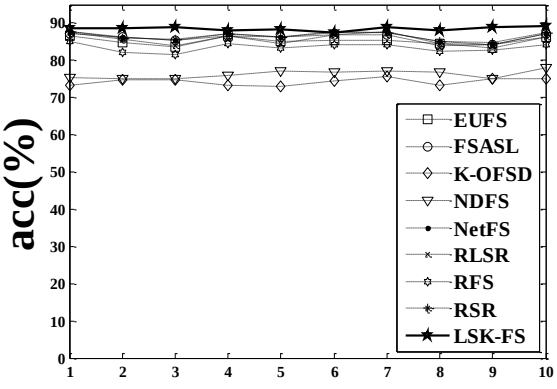
Clean



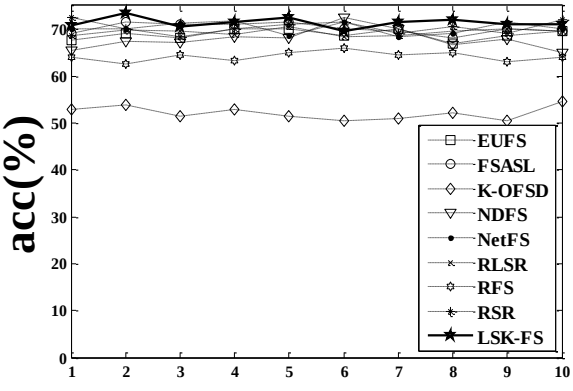
Colon



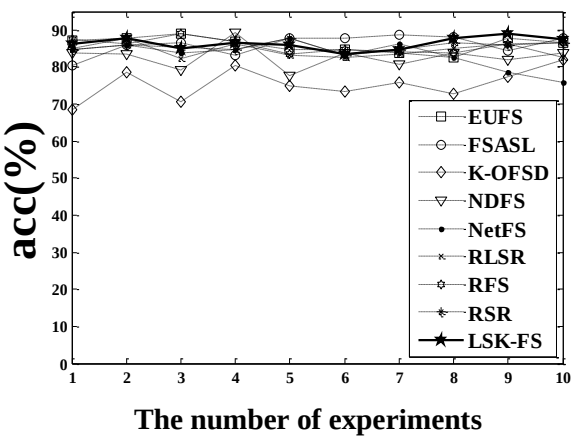
Ecoli



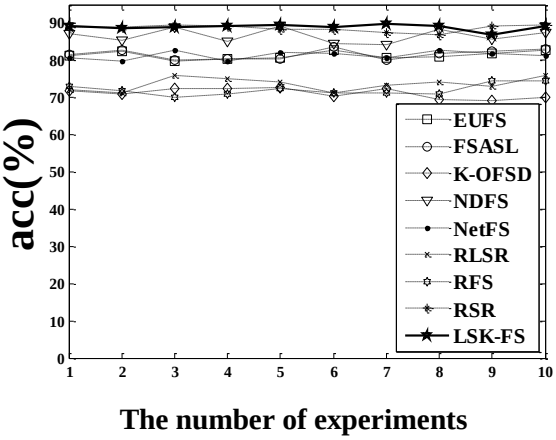
Forest



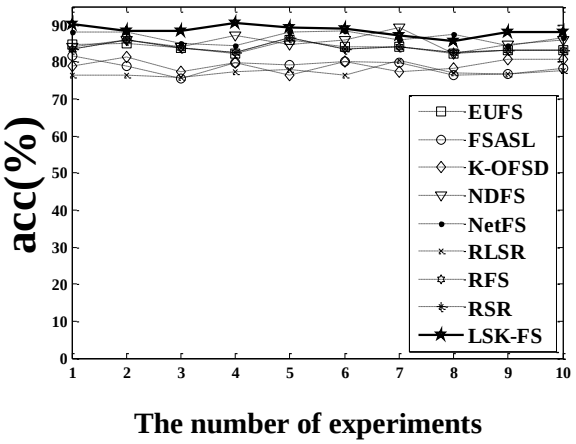
Glass



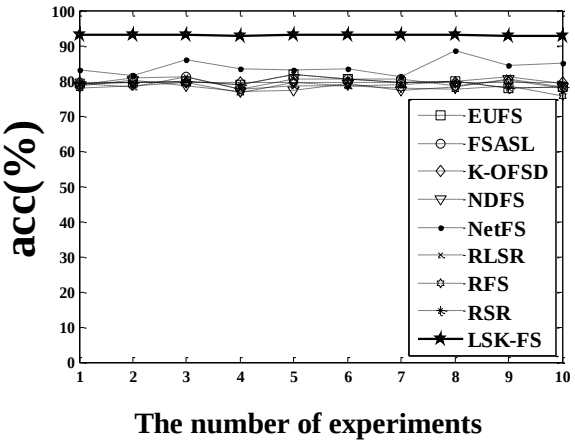
Lungdiscrete



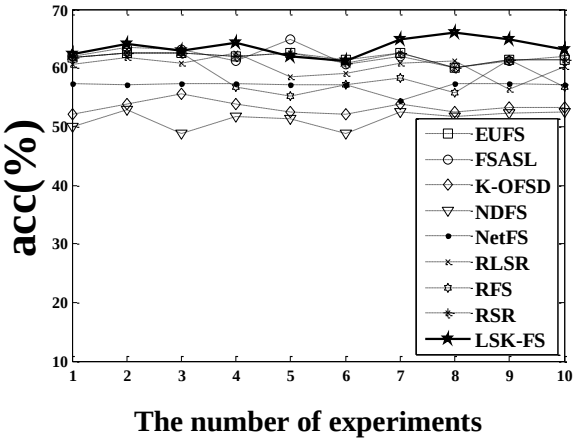
Movements



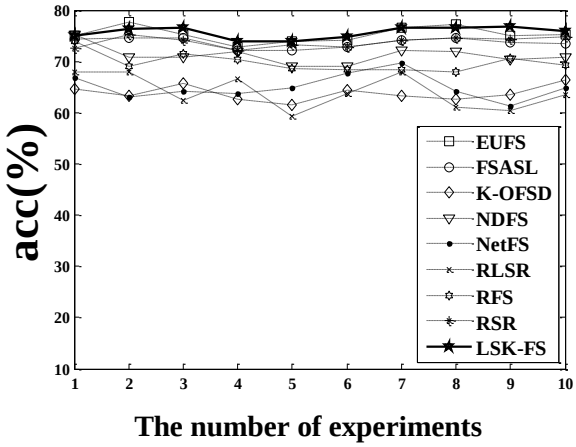
Sonar



SPECTF



Urban_land



Yale

Figure 1. Average classification accuracy of all methods for all datasets

Table 2. Average classification accuracy (acc(%))

Datasets	EUFS	FSASL	K_OFSD	NDFS	NetFS	RLSR	RFS	RSR	LSK_FS
Arrhythmia	60.52	66.71	60.74	65.57	54.20	70.53	64..02	66.33	70.96
Clean	84.61	84.83	86.53	77.82	95.78	91.53	84.64	84.77	96.13
Colon	82.88	77.07	64.60	64.43	64.38	64.42	82.48	83.17	84.14
Ecoli	84.97	86.01	69.28	82.81	84.47	78.31	85.74	85.81	86.19
Forest	85.23	86.06	74.23	76.31	86.43	85.78	83.50	86.55	88.50
Glass	68.90	70.17	52.09	67.86	70.20	69.36	64.17	70.42	71.38
Lungdiscrete	86.11	86.07	75.45	82.77	73.41	84.43	85.88	85.55	86.41
Movements	81.36	81.61	71.19	86.61	81.42	73.58	72.08	88.61	88.92
Sonar	83.81	78.67	79.11	85.43	86.66	77.26	83.91	83.91	88.48
SPECTF	79.54	79.70	79.40	78.48	83.95	79.22	78.04	79.54	92.94
Urban_land	61.78	62.05	53.25	51.27	56.96	60.20	58.82	61.72	63.56
Yale	75.27	73.69	63.78	71.22	65.04	64.05	69.76	73.81	75.63
Average value	77.92	77.72	69.14	74.22	76.08	74.89	76.09	79.18	82.77

In Table 3, we can see the average standard deviation of each algorithm on 12 data sets. The standard deviation of the proposed LSK_FS algorithm is the smallest, indicating that our algorithm has the best stability.

Table 3. Standard deviation of classification accuracy (std(%))

Datasets	EUFS	FSASL	K_OFSD	NDFS	NetFS	RLSR	RFS	RSR	LSK_FS
Arrhythmia	0.88	0.85	1.04	0.89	0.02	0.85	1.29	1.09	0.79
Clean	0.54	0.94	0.03	1.29	1.32	0.03	0.97	0.74	0.49
Colon	1.88	2.99	0.38	1.31	0.29	0.59	2.08	0.93	1.5
Ecoli	1.18	0.47	0.30	1.59	1.78	0.55	0.40	0.82	0.69
Forest	1.05	1.05	0.93	1.05	1.06	1.29	1.16	0.99	0.54
Glass	1.05	1.02	1.37	2.07	1.15	0.70	1.00	1.29	1.01
Lungdiscrete	1.92	2.52	4.07	3.03	3.43	1.82	1.67	1.21	1.60
Movements	1.04	1.31	1.33	1.76	1.06	1.65	1.44	0.80	0.72
Sonar	1.18	1.81	1.62	1.85	1.45	1.19	1.28	1.28	1.38
SPECTF	1.01	1.24	0.03	1.20	2.08	1.10	1.11	1.01	0.04
Urban_land	0.77	1.38	1.01	1.44	0.88	1.69	2.74	0.83	1.41
Yale	1.44	0.95	1.4	1.71	2.28	3.16	1.74	0.99	1.08
Average value	1.16	1.38	1.13	1.60	1.40	1.22	1.41	1.00	0.94

The LSK_FS algorithm can achieve such good results, mainly for the following two reasons: 1. Consider the similarity between data features. 2. Fully consider the nonlinear relationship between data features.

4-3- Parameter Sensitivity Analysis

We set the range of values for λ_1 and λ_2 to $\{10^{-2},10^{-1},...,10^{2}\}$. By adjusting the values of parameters λ_1 and λ_2, we show the results of our proposed algorithm in Figure 2.

As shown in Figure 2, the proposed algorithm is sensitive to adjusting parameters. Specifically, it is not very sensitive on most data sets, but it also has subtle changes. This is because the algorithm proposed by us has better robustness. λ_1 is used to control the value of $\sum_{i,j}^{d}\|x_i^T\boldsymbol{AB}-x_j^T\boldsymbol{AB}\|_2^2 s_{i,j}$, while $\sum_{i,j}^{d}\|x_i^T\boldsymbol{AB}-x_j^T\boldsymbol{AB}\|_2^2 s_{i,j}$ takes into account the nonlinear relationship of the features. λ_2 is used to control the value of $\|s_i\|_2^2$, and $\|s_i\|_2^2$ takes into account the similarity of features. Therefore, in our proposed algorithm, it is necessary to adjust the parameters.

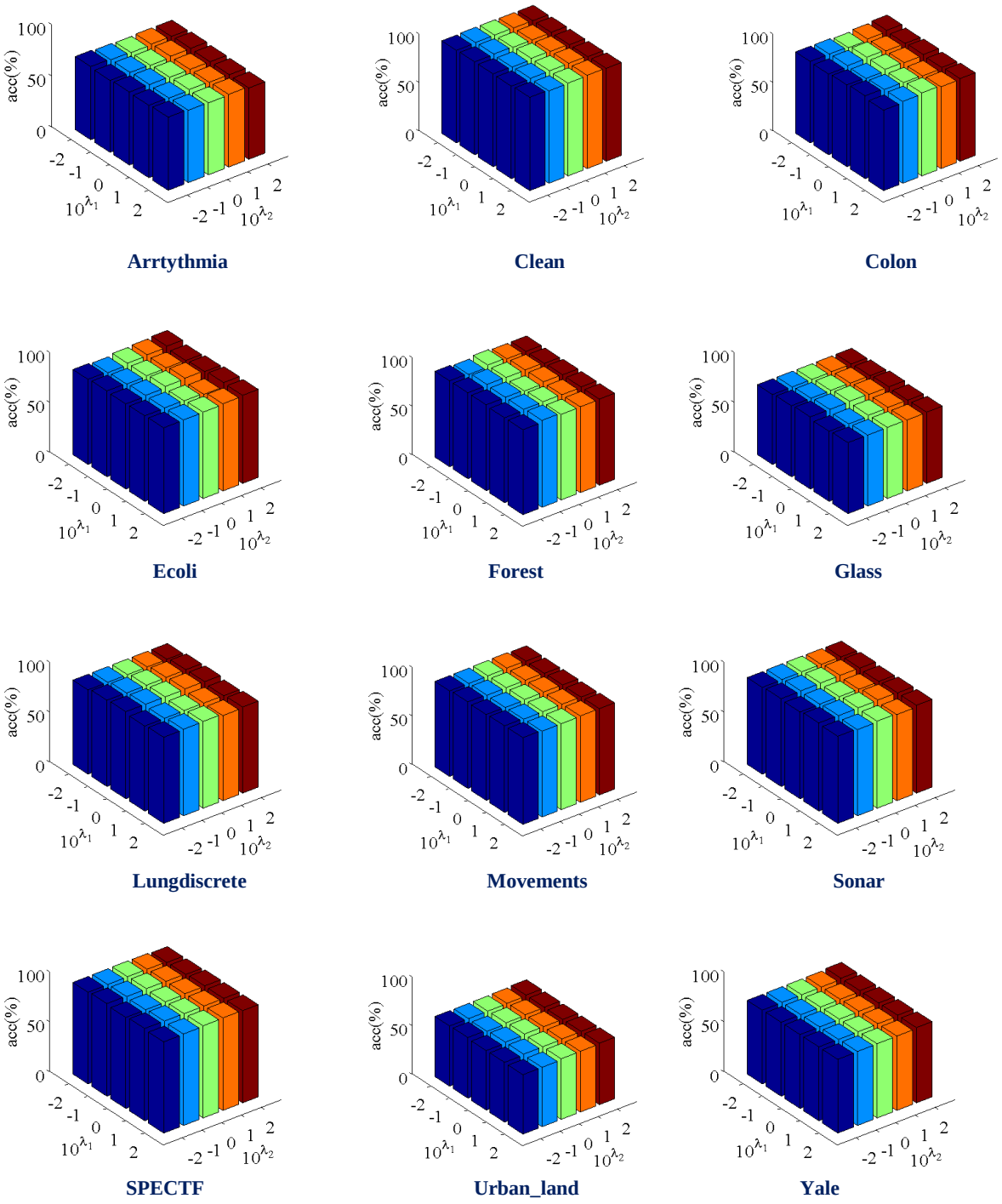


Figure 2. Accuracy of the proposed algorithm under different parameter values

4-3- Convergence Analysis

In Figure 3, we show the objective function values for each iteration of the proposed algorithm, on 12 data sets. We set the stop criteria of the proposed algorithm to $\|obj(t+1)-obj(t)\|/obj(t)\leq 10^{-4}$, where $obj(t)$ represents the value of the objective function at the *t*-th iteration. From Figure 3 we can see that: 1. The proposed optimization algorithm is convergent, which makes the value of the objective function gradually decrease in each iteration, and finally converges. 2. The proposed algorithm converges on most datasets within 20 iterations, which indicates that the proposed algorithm converges quickly.

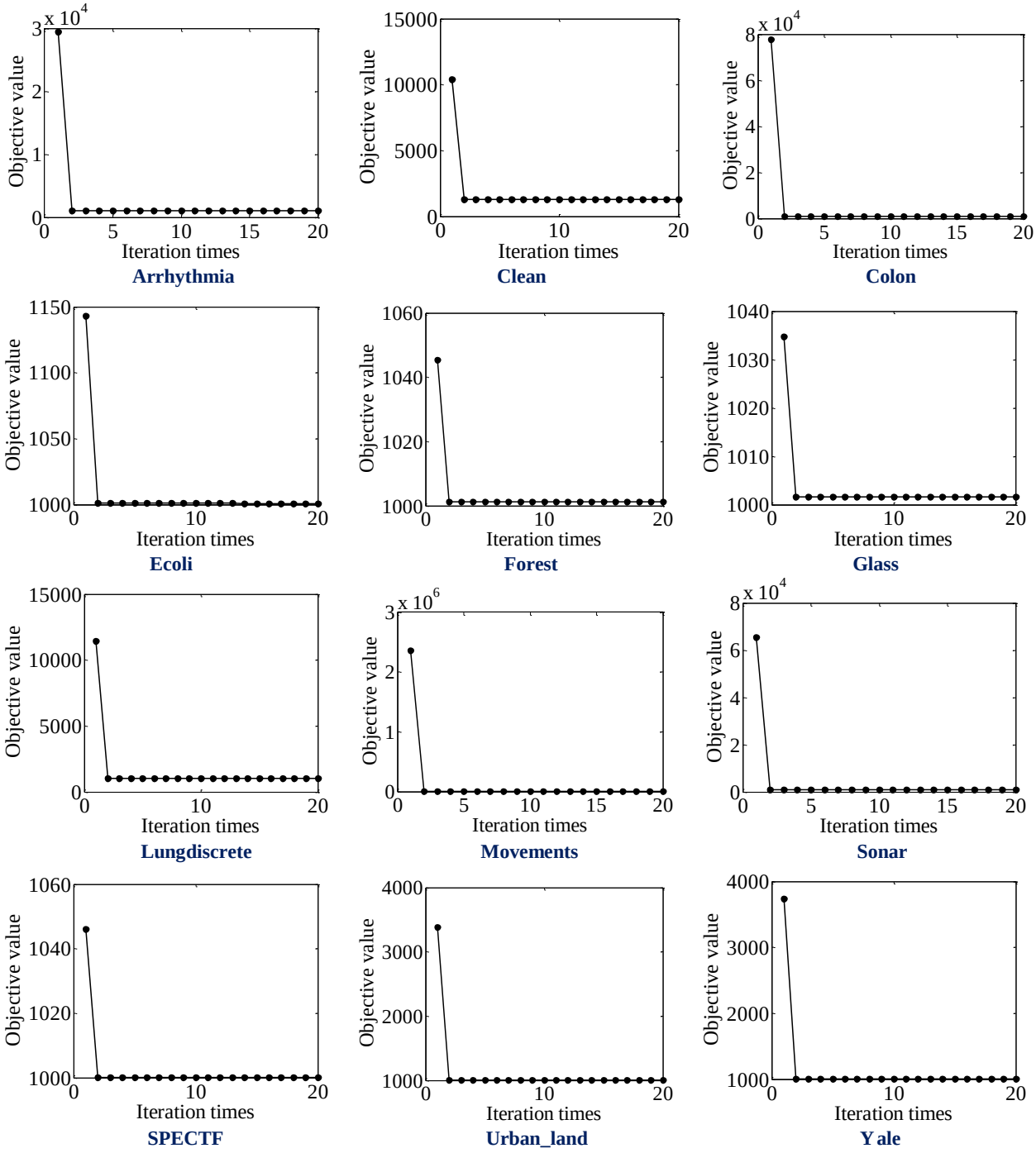


Figure 3. The convergence graph of algorithm 1 for all data sets

5- Conclusion

This article has proposed a new feature selection algorithm to remove redundant features. Specifically, the local structure learning is applied to the features to take into account the local structure and similarity of the features. Moreover, the kernel function is applied to map all the features in the high-dimensional space, so that the nonlinear relationship of the features is linearly separable in the high-dimensional space. In addition, low rank constraints are used to remove noise samples, maintain the global structure of the data and achieve robust results. Finally, the experimental results also show the superiority of our proposed algorithm. In future work, we plan to improve our algorithms through robust statistical learning.

6- Funding and Acknowledgements

This work is partially supported by the China Key Research Program (Grant No: 2016YFB1000905); the Key Program of the National Natural Science Foundation of China (Grant No: 61836016); the Natural Science Foundation of China (Grants No: 61876046, 61573270, 81701780 and 61672177); the Project of Guangxi Science and Technology

(GuiKeAD17195062); the Guangxi Natural Science Foundation (Grant No: 2015GXNSFCB139011, 2017GXNSFBA198221); the Guangxi Collaborative Innovation Center of Multi-Source Information Integration and Intelligent Processing; the Guangxi High Institutions Program of Introducing 100 High-Level Overseas Talents; Innovation Project of Guangxi Graduate Education(Grant No: YCSW2019073) and the Research Fund of Guangxi Key Lab of Multisource Information Mining & Security (18-A-01-01).

7- Conflict of Interest

The authors declare no conflict of interest.

7- References

[1] X. Zhu, H. Suk and D. Shen, "Matrix-Similarity Based Loss Function and Feature Selection for Alzheimer's Disease Diagnosis," 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 3089-3096. doi: 10.1109/CVPR.2014.395.

[2] Zhu, Xiaofeng, Zi Huang, Yang Yang, Heng Tao Shen, Changsheng Xu, and Jiebo Luo. "Self-Taught Dimensionality Reduction on the High-Dimensional Small-Sized Data." Pattern Recognition 46, no. 1 (January 2013): 215–229. doi:10.1016/j.patcog.2012.07.018.

[3] Zhu, Xiaofeng, Shichao Zhang, Zhi Jin, Zili Zhang, and Zhuoming Xu. "Missing Value Estimation for Mixed-Attribute Data Sets." IEEE Transactions on Knowledge and Data Engineering 23, no. 1 (January 2011): 110–121. doi:10.1109/tkde.2010.99.

[4] Bolón-Canedo, Verónica, Noelia Sánchez-Maroño, and Amparo Alonso-Betanzos. "Feature Selection for High-Dimensional Data." Progress in Artificial Intelligence 5, no. 2 (February 15, 2016): 65–75. doi:10.1007/s13748-015-0080-y.

[5] Ling, Charles X., Qiang Yang, Jianning Wang, and Shichao Zhang. "Decision Trees with Minimal Costs." Twenty-First International Conference on Machine Learning - ICML '04 (2004). doi:10.1145/1015330.1015369.

[6] Zhang, Shichao, Zhi Jin, and Xiaofeng Zhu. "Missing Data Imputation by Utilizing Information Within Incomplete Instances." Journal of Systems and Software 84, no. 3 (March 2011): 452–459. doi:10.1016/j.jss.2010.11.887.

[7] Nie F, Zhu W, Li X. (2016). Unsupervised feature selection with structured graph optimization[C]// Thirtieth AAAI Conference on Artificial Intelligence. AAAI Press, 2016:1302-1308.

[8] Almusallam, Naif, Zahir Tari, Jeffrey Chan, and Adil AlHarthi. "UFSSF - An Efficient Unsupervised Feature Selection for Streaming Features." Lecture Notes in Computer Science (2018): 495–507. doi:10.1007/978-3-319-93037-4_39.

[9] Xue, Wei, and Wensheng Zhang. "Online Weighted Multi-Task Feature Selection." Lecture Notes in Computer Science (2016): 195–203. doi:10.1007/978-3-319-46672-9_23.

[10] Xinwang Liu, Lei Wang, Jian Zhang, Jianping Yin, and Huan Liu ,"Global and Local Structure Preservation for Feature Selection." IEEE Transactions on Neural Networks and Learning Systems 25, no. 6 (June 2014): 1083–1095. doi:10.1109/tnnls.2013.2287275.

[11] Wan, Yuan, Xiaoli Chen, and Jinghui Zhang. "Global and Intrinsic Geometric Structure Embedding for Unsupervised Feature Selection." Expert Systems with Applications 93 (March 2018): 134–142. doi:10.1016/j.eswa.2017.10.008.

[12] Li, Yun, Jennie Si, Guojing Zhou, Shasha Huang, and Songcan Chen. "FREL: A Stable Feature Selection Algorithm." IEEE Transactions on Neural Networks and Learning Systems 26, no. 7 (July 2015): 1388–1402. doi:10.1109/tnnls.2014.2341627.

[13] Tsagris, Michail, Vincenzo Lagani, and Ioannis Tsamardinos. "Feature Selection for High-Dimensional Temporal Data." BMC Bioinformatics 19, no. 1 (January 23, 2018). doi:10.1186/s12859-018-2023-7.

[14] Zhao, Hong, Ping Wang, and Qinghua Hu. "Cost-Sensitive Feature Selection Based on Adaptive Neighborhood Granularity with Multi-Level Confidence." Information Sciences 366 (October 2016): 134–149. doi:10.1016/j.ins.2016.05.025.

[15] Wang, Wei, Yan Yan, Stefan Winkler, and Nicu Sebe. "Category Specific Dictionary Learning for Attribute Specific Feature Selection." IEEE Transactions on Image Processing 25, no. 3 (March 2016): 1465–1478. doi:10.1109/tip.2016.2523340.

[16] Sheeja, T.K., and A. Sunny Kuriakose. "A Novel Feature Selection Method Using Fuzzy Rough Sets." Computers in Industry 97;111–121 (May 2018): 111–116. doi:10.1016/j.compind.2018.01.014.

[17] Zhang, Tao, Biyun Ding, Xin Zhao, and Qianyu Yue. "A Fast Feature Selection Algorithm Based on Swarm Intelligence in Acoustic Defect Detection." IEEE Access 6 (2018): 28848–28858. doi:10.1109/access.2018.2833164.

[18] Almusallam, Naif, Zahir Tari, Jeffrey Chan, and Adil AlHarthi. "UFSSF - An Efficient Unsupervised Feature Selection for Streaming Features." Lecture Notes in Computer Science (2018): 495–507. doi:10.1007/978-3-319-93037-4_39.

[19] Xue, Wei, and Wensheng Zhang. "Online Weighted Multi-Task Feature Selection." Lecture Notes in Computer Science (2016):

195–203. doi:10.1007/978-3-319-46672-9_23.

[20] Liu X, Wang L, Zhang J, Yin Jianping and Liu Huan. “Global and Local Structure Preservation for Feature Selection.” IEEE Transactions on Neural Networks and Learning Systems 25, no. 6 (June 2014): 1083–1095. doi:10.1109/tnnls.2013.2287275.

[21] Wan, Yuan, Xiaoli Chen, and Jinghui Zhang. “Global and Intrinsic Geometric Structure Embedding for Unsupervised Feature Selection.” Expert Systems with Applications 93 (March 2018): 134–142. doi:10.1016/j.eswa.2017.10.008.

[22] Li, Yun, Jennie Si, Guojing Zhou, Shasha Huang, and Songcan Chen. “FREL: A Stable Feature Selection Algorithm.” IEEE Transactions on Neural Networks and Learning Systems 26, no. 7 (July 2015): 1388–1402. doi:10.1109/tnnls.2014.2341627.

[23] Tsagris, Michail, Vincenzo Lagani, and Ioannis Tsamardinos. “Feature Selection for High-Dimensional Temporal Data.” BMC Bioinformatics 19, no. 1 (January 23, 2018). doi:10.1186/s12859-018-2023-7.

[24] Zhao, Hong, Ping Wang, and Qinghua Hu. “Cost-Sensitive Feature Selection Based on Adaptive Neighborhood Granularity with Multi-Level Confidence.” Information Sciences 366 (October 2016): 134–149. doi:10.1016/j.ins.2016.05.025.

[25] Wang, Wei, Yan Yan, Stefan Winkler, and Nicu Sebe. “Category Specific Dictionary Learning for Attribute Specific Feature Selection.” IEEE Transactions on Image Processing 25, no. 3 (March 2016): 1465–1478. doi:10.1109/tip.2016.2523340.

[26] Sheeja, T.K., and A. Sunny Kuriakose. “A Novel Feature Selection Method Using Fuzzy Rough Sets.” Computers in Industry 97 (May 2018): 111–116. doi:10.1016/j.compind.2018.01.014.

[27] Zhang, Tao, Biyun Ding, Xin Zhao, and Qianyu Yue. “A Fast Feature Selection Algorithm Based on Swarm Intelligence in Acoustic Defect Detection.” IEEE Access 6 (2018): 28848–28858. doi:10.1109/access.2018.2833164.

[28] Zhang, Chengqi, Yongsong Qin, Xiaofeng Zhu, Jilian Zhang, and Shichao Zhang. “Clustering-Based Missing Value Imputation for Data Preprocessing.” 2006: 1081-1086 IEEE International Conference on Industrial Informatics (August 2006). doi:10.1109/indin.2006.275767.

[29] Qin, Yongsong, Shichao Zhang, Xiaofeng Zhu, Jilian Zhang, and Chengqi Zhang. “Semi-Parametric Optimization for Missing Data Imputation.” Applied Intelligence 27, no. 1 (January 18, 2007): 79–88. doi:10.1007/s10489-006-0032-0.

[30] Wang, Suhang, Jiliang Tang, and Huan Liu. "Embedded unsupervised feature selection." In Twenty-ninth AAAI conference on artificial intelligence. 2015: 470-476.

[31] Du, Liang, and Yi-Dong Shen. “Unsupervised Feature Selection with Adaptive Structure Learning.” Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD ’15 (2015). doi:10.1145/2783258.2783345.

[32] Li, Zechao, Yi Yang, Jing Liu, Xiaofang Zhou, and Hanqing Lu. "Unsupervised feature selection using nonnegative spectral analysis." In Twenty-Sixth AAAI Conference on Artificial Intelligence. 2012: 1026-1032.

[33] Li, Jundong, Xia Hu, Liang Wu, and Huan Liu. "Robust unsupervised feature selection on networked data." In Proceedings of the 2016 SIAM International Conference on Data Mining, pp. 387-395. Society for Industrial and Applied Mathematics, 2016: 387-395.

[34] Chen, Xiaojun, Guowen Yuan, Wenting Wang, Feiping Nie, Xiaojun Chang, and Joshua Zhexue Huang. “Local Adaptive Projection Framework for Feature Selection of Labeled and Unlabeled Data.” IEEE Transactions on Neural Networks and Learning Systems 29, no. 12 (December 2018): 6362–6373. doi:10.1109/tnnls.2018.2830186.

[35] Nie, Feiping, Heng Huang, Xiao Cai, and Chris H. Ding. "Efficient and robust feature selection via joint ℓ2, 1-norms minimization." In Advances in neural information processing systems, pp. 1813-1821. 2010.

[36] Zhu, Pengfei, Wangmeng Zuo, Lei Zhang, Qinghua Hu, and Simon C.K. Shiu. “Unsupervised Feature Selection by Regularized Self-Representation.” Pattern Recognition 48, no. 2 (February 2015): 438–446. doi:10.1016/j.patcog.2014.08.006.

[37] Zhou, Peng, Xuegang Hu, Peipei Li, and Xindong Wu. “Online Feature Selection for High-Dimensional Class-Imbalanced Data.” Knowledge-Based Systems 136 (November 2017): 187–199. doi:10.1016/j.knosys.2017.09.006.

[38] Nesterov, Yurii. “Nonsmooth Convex Optimization.” Introductory Lectures on Convex Optimization (2004): 111–170. doi:10.1007/978-1-4419-8853-9_3.